**IPV – Instituto Politécnico de Viseu**

**ESTGV – Escola Superior de Tecnologia e Gestão de Viseu**

**Departamento de Informática**


# Relatório de Aplicações para a Internet

**Licenciatura em Engenharia Informática**

**Realizado**

**por**

**Artur Santos pv20251**

**Luís Anselmo pv20245**

**Alexandre Moreira pv20223**

**Orientador**

**ESTGV: Steven Abrantes**

**Viseu, 2021**

**IPV – Instituto Politécnico de Viseu**

**ESTGV – Escola Superior de Tecnologia e Gestão de Viseu**

**Departamento de Informática**

# Relatório de Aplicações para a Internet

## Licenciatura em Engenharia Informática

**Realizado**

**por**

**Artur Santos pv20251**

**Luís Anselmo pv20245**

**Alexandre Moreira pv20223**

**Orientadores**

**ESTGV: Steven Abrantes**

**Viseu, 2021**

# Índice

# 1. Introdução

Neste trabalho prático iremos demonstrar tudo oque aprendemos ao longo do semestre nesta cadeira.

Neste trabalho pretende-se criar um site a nossa escolha utilizando os diversos conteúdos que nos foram lecionados ao longo do semestre. O nosso grupo decidiu que iriamos fazer um portefólio sobre todos os elementos do grupo

Neste trabalho, o utilizador pode ver o portefólio assim como algumas breves informações sobre os elementos do grupo e também comunicar com os mesmos.

# 2. HTML

## 2.1. <head>

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="assets/css/styles.css">
    <title>PORTFOLIO</title>
</head>
```

Ao longo do programa todas as páginas escritas em html têm o mesmo <*head*>, onde chamamos o ficheiro relativo a CSS e definimos o nome da página.

## 2.2. <header>

```
<body onload="startTime()">
    <!-- HEADER -->
    <header class="l-header">
        <nav class="nav bd-grid">
            <div>
                <a href="index.html" class="nav_logo">PORTFOLIO</a>
            </div>
            <div class="nav_menu" id="nav-menu">
                <ul class="nav_list">
                    <li class="nav_item"><a href="index.html" class="nav_link">Home</a></li>
                    <li class="nav_item"><a href="#" class="nav_link">Quem Somos</a></li>
                    <li class="nav_item"><a href="contacto.html" class="nav_link">Contacte-nos</a></li>
                    <div class="relogio" id="relogio"></div>
                </ul>
            </div>
            <div class="nav_toggle" id="nav-toggle">
                <ion-icon name="menu"></ion-icon>
            </div>
        </nav>
    </header>
```

Assim como no <*head*>, o <*header*> é igual para todas as páginas apenas mudando o link da própria página no menu de navegação. Na imagem a cima podemos ver que ao carregar o body, o programa automaticamente inicializa a função '*startTime*' permitindo carregar imediatamente a hora local, no seguimento de inserir um relógio que permanecerá no canto superior da página.

## 2.3. <main>

```
<main class="l-main bd-grid">
    <div class="home">
        <div class="home_information">
            <span class="home_pressent anime-text">Olá, o meu nome é</span>
            <h1 class="home_title anime-text">Artur</h1>
            <span class="home_skill anime-text">Web Developer</span>
            <div>
                <a href="contacto.html" class="home_button anime-text">Contactar</a>
            </div>
        </div>
        <div class="home_information">
            <span class="home_pressent anime-text">Olá, o meu nome é</span>
            <h1 class="home_title anime-text">Alexandre</h1>
            <span class="home_skill anime-text">Web Developer</span>
            <div>
                <a href="contacto.html" class="home_button anime-text">Contactar</a>
            </div>
        </div>
        <div class="home_information">
            <span class="home_pressent anime-text">Olá, o meu nome é</span>
            <h1 class="home_title anime-text">Luís</h1>
            <span class="home_skill anime-text">Web Developer</span>
            <div>
                <a href="contacto.html" class="home_button anime-text">Contactar</a>
            </div>
        </div>
        <div class="home_img">
            <div class="mySlides fade">
                <img src="assets/imagens/work7.jpg">
            </div>
            <div class="mySlides fade">
                <img src="assets/imagens/work8.jpg">
            </div>
            <div class="mySlides fade">
                <img src="assets/imagens/work9.jpg">
            </div>
        </div>
        <div class="home_social">
            <a href="https://moodle.estgv.ipv.pt/course/view.php?id=4280" class="footer_icon">
                <ion-icon name="school" class="home_social-icon"></ion-icon>
            </a>
            <a href="https://github.com/ArturSantos23/Projeto-AI1" class="footer_icon">
                <ion-icon name="logo-github" class="home_social-icon"></ion-icon>
            </a>
            <a href="https://github.com/ArturSantos23/Projeto-AI1/blob/main/Enunciado_Projecto_Pratico.pdf"
                class="footer_icon">
                <ion-icon name="help-circle-outline" class="home_social-icon"></ion-icon>
            </a>
        </div>
```

Na imagem a cima podemos ver o conteúdo existente no <*main*> na pagina inicial do site. Neste ponto podemos observar a existência de '*sliders*', quer para imagens quer para a informação relativa aos criadores. Para alem dos '*sliders*' referidos, temos também alguns icons que permitem o utilizador visitar alguns sites relacionados com a página e os seus criadores.

```
<main class="l-main bd-grid">
    <div class="home">
        <section class="contact section" id="contact">
        </section>
    </div>
</main>
```

No caso da página de contactos, todo o código desenvolvido esta presente no ficheiro relativo a ‘react’ (react.jsx)

```
<main class="div_cards">
    <div class="card">
        <img src="assets/imagens/Artur.png" alt="Avatar" style="width:100%">
        <div class="container">
            <h4><b>Artur Santos</b></h4>
            <p>Estudante Universitário</p>
        </div>
    </div>
    <div class="card2">
        <img src="assets/imagens/Alexandre.jpg" alt="Avatar" style="width:100%">
        <div class="container">
            <h4><b>Alexandre Moreira</b></h4>
            <p>Estudante Universitário</p>
        </div>
    </div>
    <div class="card3">
        <img src="assets/imagens/Luis.jpg" alt="Avatar" style="width:100%">
        <div class="container">
            <h4><b>Luís Anselmo</b></h4>
            <p>Estudante Universitário</p>
        </div>
    </div>
</main>
```

Finalmente na última página, ‘*about*’, o utilizador pode ver mais alguns detalhes os criadores do site.

## 2.4. <footer>

```
<footer class="footer">
    <p class="footer_title">Artur Santos</p>
    <div class="footer_social">
        <a href="https://www.facebook.com/profile.php?id=100014925133985" class="footer_icon">
            <ion-icon name="logo-facebook"></ion-icon>
        </a>
        <a href="https://www.instagram.com/artur_santos_23/" class="footer_icon">
            <ion-icon name="logo-instagram"></ion-icon></i>
        </a>
        <a href="https://twitter.com/Artur_Santos23" class="footer_icon">
            <ion-icon name="logo-twitter"></ion-icon>
        </a>
    </div>
    <p class="footer_copy">&#169; Artur Santos. All rigths reserved</p>
</footer>
<footer class="footer">
    <p class="footer_title">Alexandre Moreira</p>
    <div class="footer_social">
        <a href="https://www.facebook.com/profile.php?id=100006307127886" class="footer_icon">
            <ion-icon name="logo-facebook"></ion-icon>
        </a>
        <a href="https://www.instagram.com/alex.moreira0202/" class="footer_icon">
            <ion-icon name="logo-instagram"></ion-icon></i>
        </a>
        <a href="https://twitter.com/MoreiraValejo" class="footer_icon">
            <ion-icon name="logo-twitter"></ion-icon>
        </a>
    </div>
    <p class="footer_copy">&#169; Alexandre Moreira. All rigths reserved</p>
</footer>
<footer class="footer">
    <p class="footer_title">Luís Anselmo</p>
    <div class="footer_social">
        <a href="https://www.facebook.com/anselmoheh/" class="footer_icon">
            <ion-icon name="logo-facebook"></ion-icon>
        </a>
        <a href="https://www.instagram.com/luis_anselmoheh.14/" class="footer_icon">
            <ion-icon name="logo-instagram"></ion-icon></i>
        </a>
        <a href="https://twitter.com/anselmoheh" class="footer_icon">
            <ion-icon name="logo-twitter"></ion-icon>
        </a>
    </div>
    <p class="footer_copy">&#169; Luís Anselmo. All rigths reserved</p>
</footer>
```

Como pode ser visto na imagem apresentada, o '*footer*' apresenta o nome dos criadores do site assim como as redes sociais dos mesmos. A rotação dos utilizadores apresentados esta diretamente ligada aos '*sliders*' anteriores.

## 2.5. Overlay

```
<!-- OVERLAY -->
<div class="overlay first"></div>
<div class="overlay second"></div>
<div class="overlay third"></div>
```

Com ajuda de alguma pesquisa e com o objetivo de deixar a página inicial mais apelativa, ao iniciar o site o utilizador será recebido por alguns *overlays* animados.

## 2.6. Scripts Finais

```
<!-- GSAP (Animações)-->
<script src="https://cdnjs.cloudflare.com/ajax/libs/gsap/3.2.4/gsap.min.js" crossorigin></script>

<!-- ICONS -->
<script type="module" src="https://unpkg.com/ionicons@4.5.10-0/dist/ionicons/ionicons.esm.js"></script>

<!-- MAIN (TEM DE APARECER NO FINAL DO CODIGO HTML)-->
<script src="assets/js/main.js" async></script>
<script type="text/babel" src="assets/js/react.jsx" crossorigin></script>
<script src="https://unpkg.com/react@17/umd/react.development.js" crossorigin></script>
<script src="https://unpkg.com/react-dom@17/umd/react-dom.development.js" crossorigin></script>
<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
```

Finalmente, no fim de todas as páginas html são apresentados diversos scripts que relativos à componente JavaScript e React, assim com utilização de animações (GSAP) e de icons (ionicons)

# 3. JavaScript

```
const showMenu = (toggleId, navId) => {
  const toggle = document.getElementById(toggleId),
    nav = document.getElementById(navId);

  if (toggle && nav) {
    toggle.addEventListener("click", () => {
      nav.classList.toggle("show");
    });
  }
};

showMenu("nav-toggle", "nav-menu");
```

A função apresentada em cima é relativa ao menu de navegação. Quando o utilizador visualiza o website num dispositivo mais pequeno, o menu da página é substituído por um “hamburger menu”.

```
function startTime() {
  const today = new Date();
  let h = today.getHours();
  let m = today.getMinutes();
  let s = today.getSeconds();
  m = checkTime(m);
  s = checkTime(s);
  document.getElementById("relogio").innerHTML = h + ":" + m + ":" + s;
  setTimeout(startTime, 1000);
}

function checkTime(i) {
  if (i < 10) {
    i = "0" + i;
  } // adiciona 0 aos numeros < 10
  return i;
}
```

A função mostrada na figura acima é a função responsável pelo relógio que é carregada no body de cada página (referido no header).

```
var slideIndex1 = 0;
var slideIndex2 = 0;
var slideIndex3 = 0;
showSlides();
showSlides_Div();
showSlides_Footer();

function showSlides_Div() {
  var i;
  var slidesDiv = document.getElementsByClassName("home_information");
  for (i = 0; i < slidesDiv.length; i++) {
    slidesDiv[i].style.display = "none";
  }
  slideIndex1++;
  if (slideIndex1 > slidesDiv.length) {
    slideIndex1 = 1;
  }
  slidesDiv[slideIndex1 - 1].style.display = "block";
  setTimeout(showSlides_Div, 5000); // Muda div de 5 em 5 segundos
}

function showSlides() {
  var i;
  var slides = document.getElementsByClassName("mySlides");
  for (i = 0; i < slides.length; i++) {
    slides[i].style.display = "none";
  }
  slideIndex2++;
  if (slideIndex2 > slides.length) {
    slideIndex2 = 1;
  }
  slides[slideIndex2 - 1].style.display = "block";
  setTimeout(showSlides, 5000); // Muda imagem(Home) de 5 em 5 segundos
}

function showSlides_Footer() {
  var i;
  var slidesFooter = document.getElementsByClassName("footer");
  for (i = 0; i < slidesFooter.length; i++) {
    slidesFooter[i].style.display = "none";
  }
  slideIndex3++;
  if (slideIndex3 > slidesFooter.length) {
    slideIndex3 = 1;
  }
  slidesFooter[slideIndex3 - 1].style.display = "block";
  setTimeout(showSlides_Footer, 5000); // Muda footer de 5 em 5 segundos
}
```

Todas as funções apresentadas acima são responsáveis pelo funcionamento dos *slideshows* das imagens assim como do texto da home page e o footer.

```
/*----- ANIMACOES -----*/
// INTRO OVERLAY
gsap.to(".first", 1.5, { delay: 0.5, top: "-100%", ease: Expo.easeInOut });
gsap.to(".second", 1.5, { delay: 0.7, top: "-100%", ease: Expo.easeInOut });
gsap.to(".third", 1.5, { delay: 0.9, top: "-100%", ease: Expo.easeInOut });

// SLIDER
gsap.from(".home_img", { opacity: 0, duration: 2, delay: 2, x: 60 });

// HOME
gsap.from(".home_information", { opacity: 0, duration: 3, delay: 2.3, y: 25 });
gsap.from(".anime-text", {
  opacity: 0,
  duration: 3,
  delay: 2.3,
  y: 25,
  ease: "expo.out",
  stagger: 0.3,
});

// NAV ITENS
gsap.from(".nav_logo", {
  opacity: 0,
  duration: 3,
  delay: 3.2,
  y: 25,
  ease: "expo.out",
});
gsap.from(".nav_item", {
  opacity: 0,
  duration: 3,
  delay: 3.2,
  y: 25,
  ease: "expo.out",
  stagger: 0.2,
});
```

```
// SOCIAL
gsap.from(".home_social-icon", {
  opacity: 0,
  duration: 3,
  delay: 4,
  y: 25,
  ease: "expo.out",
  stagger: 0.2,
});

// FOOTER
gsap.from(".footer", {
  opacity: 0,
  duration: 3,
  delay: 4,
  y: 25,
  ease: "expo.out",
  stagger: 0.2,
});
```

JavaScrip associado às animações (GSAP).

Ver mais informações no final do relatório em “documentação”

# 4. React

```
const form = React.createElement;
class NameForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = { name: "" , reason: "", email: "", tele: "", info: ""};
  }
```

A partir do inicio do programa podemos ver que iniciamos um criador de elemento, assim como a class principal no caso do nosso programa. Apos definirmos a class, inicializamos o construtor onde definimos os *Hoocks* que nos permitem usar *state* sem necessitar de escrever outra class. Em seguimento definimos as variáveis que vamos utilizar para guardar os dados sobre o contacto.

```
submitFunc = (event) => {
  alert("Dados foram submetidos");
  event.stopPropagation();
  let form = {
    "name": this.state.name,
    "reason": this.state.reason,
    "email": this.state.email,
    "tele": this.state.tele,
    "info": this.state.info
  };
  localStorage.setItem("Form",JSON.stringify(form));
};
```

De seguida, começamos a nossa primeira e única função *submitFunc*. Nesta função definimos uma lista com todas as variáveis que iremos utilizar, assim como a alocação dessa mesma lista no armazenamento local.

```
render() {
  return React.createElement(
    "div",
    { className: "home_information" },
    React.createElement(
      "form",
      { className: "contact__form", onSubmit: (event) => this.submitFunc(event) },
      React.createElement(
        "span",
        { className: "home_pressent anime-text" },
        "Nome"
      ),
      React.createElement("input", {
        className: "contact_input",
        type: "text",
        required: true,
        onChange: (event) => {this.setState({ name: event.target.value })},
      }),
```

```
React.createElement(
  "span",
  { className: "home_pressent anime-text" },
  "Motivo"
),
React.createElement("input", {
  className: "contact_input",
  type: "text",
  required: true,
  onChange: (event) => {this.setState({ reason: event.target.value })},
}),
```

```
React.createElement(
  "span",
  { className: "home_pressent anime-text" },
  "Email"
),
React.createElement("input", {
  className: "contact_input",
  type: "text",
  required: true,
  onChange: (event) => {this.setState({ email: event.target.value })},
}),
```

```
React.createElement(
  "span",
  { className: "home_pressent anime-text" },
  "Telemóvel"
),
React.createElement("input", {
  className: "contact_input",
  type: "text",
  required: true,
  onChange: (event) => {this.setState({ tele: event.target.value})},
}),
```

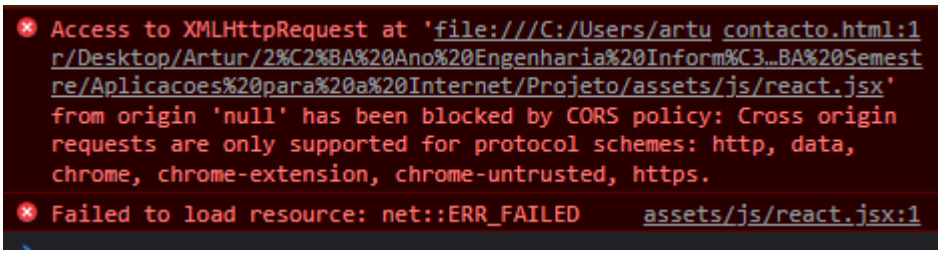
```
React.createElement(
  "span",
  { className: "home_pressent anime-text" },
  "Informação"
),
React.createElement("textarea", {
  className: "contact_input",
  type: "text",
  required: true,
  onChange: (event) => {this.setState({ info: event.target.value })},
}),
React.createElement("input",
  {
    className: "contact_button home_button anime-text",
    type: "submit",
    value: "Enviar",
  },
)
```

Nas imagens seguintes podemos ver os diversos *inputs* que têm como função registar e guardar os dados que o utilizador coloca na área de contacto assim como o botão final que permite ao utilizador submeter o formulário.

```
const domContainer = document.querySelector("#contact");
ReactDOM.render(form(NameForm), domContainer);
```

Nesta imagem final, podemos ver por fim o construtor que faz com que todo o código presente em react possa ser renderizado na nossa página *HTML*.

**Nota:** Devido a utilização de um ficheiro **.jsx** local, o react apenas funcionará se o site estiver diretamente ligado a um servidor online (use por exemplo a extensão “Live Server” do Visual Studio Code)




Após alguma pesquisa, verificámos que este erro é bastante comum quando se está na fase de desenvolvimento de um website em ambiente local. O facto de estarmos a fazer um "request" da "data" de um ficheiro externo JavaScript, no caso, o formulário, faz ativar este erro, basicamente bloqueando o acesso ao ficheiro react.jsx.

Como solução, o website deve ser acedido através de um servidor (http request). Use por exemplo a extensão "Live Server" do Visual Studio Code.

De qualquer maneira, o Website está perfeitamente operacional dentro das normas.

# Cross-Origin Resource Sharing (CORS)

CORS - Cross-Origin Resource Sharing (Compartilhamento de recursos com origens diferentes) é um mecanismo que usa cabeçalhos adicionais HTTP para informar a um navegador que permita que um aplicativo Web seja executado em uma origem (domínio) com permissão para acessar recursos selecionados de um servidor em uma origem distinta. Um aplicativo Web executa uma **requisição *cross-origin* HTTP** ao solicitar um recurso que tenha uma origem diferente (domínio, protocolo e porta) da sua própria origem.

Um exemplo de requisição *cross-origin*: o código JavaScript *frontend* de um aplicativo web disponível em `http://domain-a.com` usa `XMLHttpRequest` para fazer uma requisição para `http://api.domain-b.com/data.json`.

Por motivos de segurança, navegadores restringem requisições *cross-origin* HTTP iniciadas por scripts. Por exemplo, `XMLHttpRequest` e Fetch API seguem a política de mesma origem (*same-origin policy*). Isso significa que um aplicativo web que faz uso dessas APIs só poderá fazer solicitações para recursos de mesma origem da qual o aplicativo foi carregado, a menos que a resposta da outra origem inclua os cabeçalhos CORS corretos.

# Quem deve ler este artigo?

Todos, realmente.

Este artigo destina-se a administradores da Web, desenvolvedores de servidores e desenvolvedores front-end. Os navegadores modernos lidam com os componentes do lado cliente em compartilhamento entre origens, incluindo cabeçalhos e aplicação de políticas. Mas esse novo padrão significa que os servidores precisam lidar com novos cabeçalhos de requisição e resposta. Outro artigo para desenvolvedores de servidores que discutem compartilhamento *cross-origin* a partir de uma perspectiva de servidor (com fragmentos de código PHP), pode ser uma leitura complementar.

Ver mais informação em "documentação".

# 5. Conclusão

Após diversas horas de trabalho e pesquisa, conseguimos realizar todas as funções que nos foram pedidas.

Este programa, apesar do grau relativamente alto de conhecimento necessário para o desenvolver, acabou por ser um trabalho interessante de realizar, obrigando-nos a utilizar algumas funcionalidades ainda por explorar assim como pensar de maneiras inovadoras.

Este projeto serviu essencialmente para enriquecer o nosso conhecimento a nível de *CSS* mas sobretudo nível de *React* e *JavaScript*, sendo assim um trabalho que nos possibilitou ter uma visão mais ampla relativamente ao processo de desenvolvimento Web.

# 6. Documentação


- https://reactjs.org/docs/forms.html
- https://greensock.com/gsap/
- https://ionic.io/ionicons
- https://developer.mozilla.org/pt-BR/docs/Web/HTTP/CORS